Energy-Efficiency Evaluation of Computation Offloading in Personal Computing

Yongpil Yoon, Georgia Sakellari(✉), Richard J. Anthony, and Avgoustinos Filippoupolitis

Department of Computing and Information Systems, University of Greeenwich, London, UK
{yongpil.yoon,g.sakellari,r.j.anthony,a.filippoupolitis}@gre.ac.uk

Abstract. Cloud computing has become common practice for a wide variety of user communities. Yet, the energy efficiency and end-to-end performance benefits of cloud computing are not fully understood. Here, we focus specifically on the trade-off between local power saving and increased execution time when work is offloaded from a user's PC to a cloud environment. We have set up a 14-node private cloud and have executed a variety of applications with different processing demands. We have measured the energy cost at the level of the individual user's PC, at the level of the cloud, as well as at the two combined, contrasted to the execution time for each application when running on the PC and when running on the cloud. Our results indicate that the tradeoff between energy cost and performance differs considerably between applications of different types. In most cases investigated, the total increase in energy consumption, incurred by running that additional application, was reduced significantly. This shows that research on using cloud computing as a means to reduce the overall carbon footprint of IT is warranted. Of course, the energy gains were more pronounced for energy-selfish users, who are only interested in reducing their own carbon footprint, but these savings came at the expense of performance, with execution time increase ranging from 1 % to 84 % for different applications.

Keywords: Cloud · Computation offloading · Energy · Performance · OpenStack

1 Introduction

Cloud computing has become a common paradigm for computational resource provision. This paper investigates the viability of computation offloading to a cloud for personal computers (PCs) with regard to reducing energy costs. In other words, can computation offloading reduce the amount of required energy for a PC to complete certain tasks? And what is the overall energy consumed by the PC and the cloud in this case?

T. Czachórski et al. (Eds.): ISCIS 2016, CCIS 659, pp. 163–171, 2016.
DOI: 10.1007/978-3-319-47217-1_18

2 Related Work

Computation offloading means executing certain tasks on more resourceful computers which are not in the user's immediate computing environment, so as to: (1) reduce energy consumption of the user's computing device, and/or (2) improve the performance of computation. Computation offloading first began and has been studied mainly for mobile devices [1–5] because of the noticeable difference in computation power between mobile devices and cloud servers [6]. Performance difference between PCs and computing resources from cloud providers is often negligible and sometimes PCs outperform cloud computing resources. Although resources from clouds can be massively scalable, it may not be cost-effective depending on factors such as the type of tasks to offload, required amount of data transmission, acceptable latency etc. [7,8]. Therefore, it is important to know under what circumstances offloading is beneficial for PCs.

For mobile devices, proposed techniques may differ slightly in architectures or implementations but all share the same fundamental idea, that a mobile device can stay idle or compute less by offloading parts of program code to the cloud. Most implementations, such as Phone2Cloud [9], Cuckoo [10], COMET [11] and MAUI [12], focus on identifying tasks that can be offloaded at runtime and how this can be achieved. Recently, other perspectives of computation offloading, such as energy consumption, have been investigated. For example, the energy cost of additional communication for offloading has been addressed in [13] in order to make more energy-efficient offloading decisions in cellular networks. Computation offloading as a service for mobile devices has been suggested by [14] to bridge the gaps between the offloading demands of mobile devices and the general computing resources, such as VMs, provided by commercial cloud providers. Energy-aware scheduling of the executions of offloaded computation into the cloud has been studied in [15].

3 Experimental Methodology

We have chosen to scope our initial investigation around the energy usage considered in isolation to provide an important baseline for further work, which will take into account additional aspects including the energy cost of network communication and the additional latency of the transfers. To evaluate whether computation offloading is beneficial for PCs in terms of power consumption, we have conducted experiments using a real world private cloud. In our experiments, computation is offloaded at the application level which means the entire execution of application software was offloaded to the cloud rather than offloading some parts of computation (function/method level) like existing offloading techniques for mobile devices, e.g., MAUI - method level (RPC-like) [12], Cuckoo - method level (RMI-like) [10]. Different applications which require different amounts of computation were run both locally on a PC and remotely on a VM created in our private cloud. In the case of offloading, the VM ran the application and sent the results back to the PC or saved resulting files in the cloud when completed.

The total execution time of each application was measured as well as the power loads (Wattage) of the PC and Cloud servers during this execution time, at one-second intervals.

The experiments were conducted on a Dell Optiplex 7010 desktop machine running the Linux operating system (Ubuntu 14.04.1). The PC has Intel Core i5-3550 3.30 GHz (Quad core), 16 GB DDR3 1600 MHz memory, and 750 GB SATA-II hard drive. The power-management configurations of the PC and the OS were not changed from their default settings, e.g., sleep, hibernate, disk spin-down configurations. It was possible that the screen timeout occurs in the PC while waiting for the completion of remote execution but the power consumed by its display (monitor) was not measured. Also, the applications executed in the cloud sent the current progress of computation back to the PC after the execution had finished.

Our cloud testbed was a private *OpenStack*[1] cloud infrastructure consisting of 14 machines, each with 4-core Intel Xeon E5-2407 2.20 GHz, 48 GB DDR3 1333 MHz ECC registered memory, and 500 GB SCSI hard drive. A virtual machine with 4 virtual cores (vCPU), 8 GB memory, and 40 GB disk space was used to run the offloaded computations. There was no background traffic in the cloud during our experiments. In order to measure the power consumption of the PC a *Watts up? .Net* energy meter[2] was used. It can measure wattage to the nearest tenth of a watt with an accuracy of ±1.5 %. The meter logged the power load of the PC at 1 s intervals during the executions.

In our experiments, the computation power used by a VM in the cloud is very similar to (but slightly lower than) the user PC's. If a more powerful VM is used, our results might be different. We plan to expand our experiments to investigate the effect that the different VM configurations and PC specifications have in both the introduced power consumption and the performance of each application. However, to put things into context, the VM used is considered quite large for cloud providers. For example, Microsoft Azure considers VM instances with 4 virtual cores and 7 GB RAM as large and VM instances with 8 cores and 14 GB RAM as extra large[3]. A more powerful VM than the one we used will cost considerably more to the PC user, neutralising at least any financial benefit of the corresponding energy savings. The cost of the PC user to access the cloud is an aspect that we do not take into account here, but will also consider in the next steps of our research.

We have chosen four different applications for our experiments with the primary criterion that they are computationally intensive. All four were executed with a multithreading/multiprocessing option apart from *SCID vs. PC* which runs only on a single core. *SCID vs. PC* is a chess toolkit, which requires continuous data transmission for drawing its graphical user interface when run remotely. We ran chess engine vs. chess engine tournament which requires computation

[1] http://www.openstack.org.
[2] https://www.wattsupmeters.com.
[3] https://azure.microsoft.com/en-us/documentation/articles/cloud-services-sizes-specs/.

for searching through databases. *avconv* is an open source video and audio converting program. It is a command line program and takes video or audio files as its input and writes converted files to the disk. Video transcoding involves heavy computation as well as constant read and write to a disk is required. A 1080 p 30 fps video file of 886 MB size encoded using x264 codec was used as input data and the video was converted to a h264 mp4 file. *pi_mp.py* is a multi-threaded python implementation of π estimation using Monte Carlo method. 200 million random points were used to estimate π in each execution. It requires repetitive arithmetic calculations and a large amount of memory. *Blender* is op, featuring 3D modelling, video editing, camera object tracking, etc. In our experiments, a demo file provided by blender, called BMW benchmark, was rendered from command line. The output of the rendering is a JPEG file.

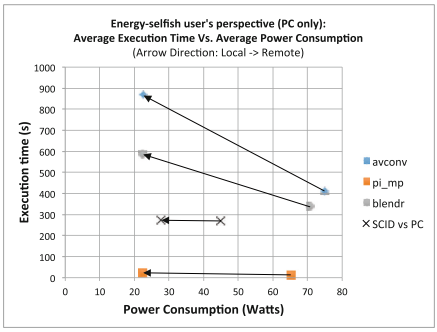
The results of the executions were sent back to the PC if it was simply text output, but if an application needed to write a file, that was saved in the cloud (in the VM where the application was executed) and thus the execution time we measured in the latter case does not include the transmission time of the resulting files. Neither the PC nor the VM in the cloud performed any other user-level activity during our experiment. There is some natural variance in the power usage of the cloud infrastructure, comprising as it does 8 compute nodes in a rack, subject to temperature fluctuations. We have found that this variation was in the worst case 3.2 %. To reduce the impact of noise in the measured cloud power usage, each application run was repeated 10 times and the average values are used in the results presented here.

4 Experimental Results on Power Consumption Vs. Performance Tradeoff

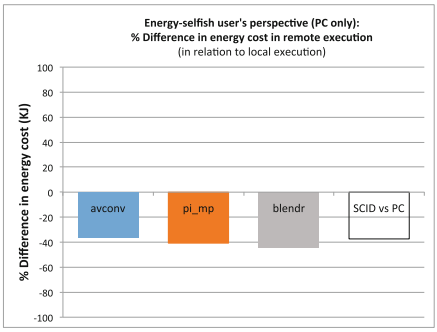
To investigate the effect of computation offloading on the energy consumption of PCs, we focus on the nature of the tradeoff between power consumption and performance. For the latter, we use the total execution time for each application, measured experimentally when running locally and when offloaded to the cloud. We have also calculated the energy consumption of the PC and the cloud (power consumption $\times$ execution time) during the executions.

4.1 Power Consumption and Performance

First we established a baseline power consumption for the PC and likewise for the cloud. The cloud required 1036.00 W on average when IDLE while the PC required only 22.23 W when IDLE. The cloud requires much more power compared to the PC since it has more machines which are power-hungrier than the PC. Obviously, the PC requires noticeably less power while simply waiting for the cloud to finish the execution, than when running applications locally. The part a's (left column) of Figs. 1, 2, 3 and 4 show the execution time vs. power consumption tradeoff. When only one core is used, about 40 % less power is required ("SCID vs PC") and when four cores are used, nearly 70 % less power is required

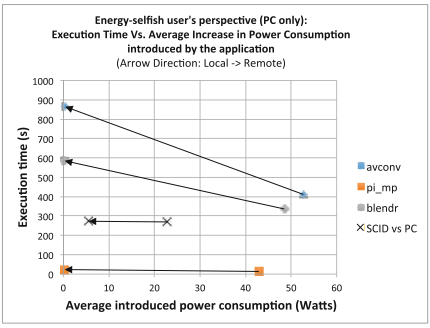


(a) (a) Average Execution Time Vs. Average Power Consumption

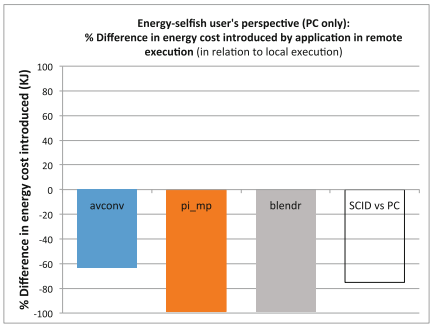


(b) (b) % Difference in PC energy cost in remote execution (in relation to local execution)

Fig. 1. Energy-selfish user's perspective (PC only)



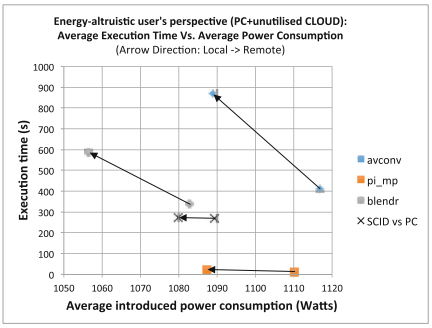
(a) (a) Execution Time Vs. Average **Increase** in Power Consumption introduced by the application



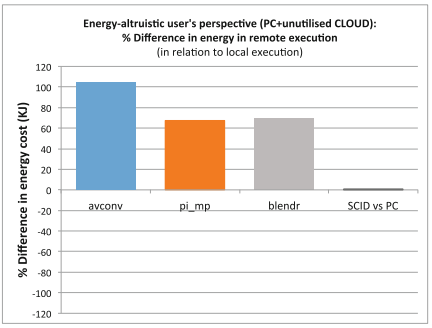
(b) (b) % Difference in PC energy cost **introduced** by an application in remote execution (in relation to local execution)

Fig. 2. Energy-selfish user's perspective (PC only): **Increases** introduced by the applications in remote operation

on average, but if seen in isolation, this is misleading. The average power load only represents the power consumption per unit time and thus, the total amount of energy consumed by each application depends on the execution time, as seen in the part b's (right column) of Figs. 1, 2, 3 and 4. The cloud required 1054.47 W of power on average during the executions. However, the introduced power load by the executions of the PC (the difference between the average power load when applications are running and when IDLE) was 41.63 W on average, while the average introduced power load in the cloud was only 18.47 W. When computation was offloaded almost all applications took much longer (up to 84 % longer) to finish certain tasks, although the VM in the cloud has the same number of

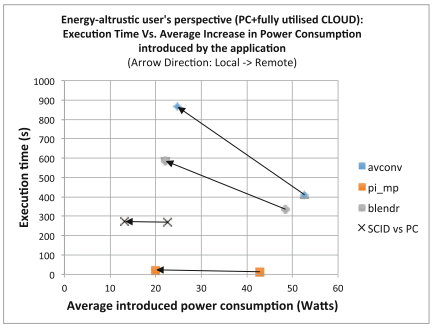


(a) (a) Average Execution Time Vs. Average Power Consumption

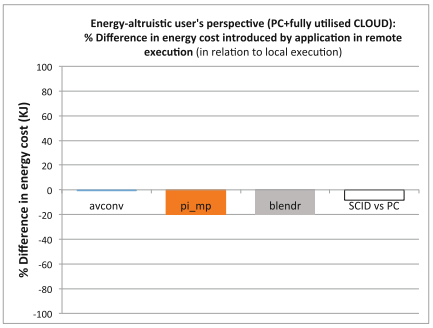


(b) (b) % Difference in total energy cost in remote execution (in relation to local execution)

Fig. 3. Energy-altruistic user's perspective (PC+CLOUD)



(a) (a) Average power consumption of PC and Cloud **introduced** by each application



(b) (b) % Difference in energy cost **introduced** by an application in remote execution (in relation to local execution)

Fig. 4. Energy-altruistic user's perspective (PC+CLOUD): **Increases** introduced by the applications in remote operation

processors as the PC. The additional end-to-end time includes network transfer latency, but this was very low because of the small amount of data needed to be transmitted. Any execution time increases were mainly due to the lower computing power of the VM in the cloud (vCPUs vs. real CPUs). Although less power is required per unit time when computation is offloaded, the total amount of energy required increases in proportion to the execution time.

4.2 Energy Consumption

The part b's (right column) of Figs. 1, 2, 3 and 4 show the percentage of the energy difference consumed on average by each application over 10 runs each, both from the PC user perspective and the total (PC+cloud) perspective.

Based on our results, the energy Vs. performance tradeoff introduced by computation offloading differs considerably depending on the application and on the perspective taken. We can broadly classify energy-conscious users as either "energy-selfish users", who are interested only in reducing the energy cost of their own PCs, versus "energy-altruistic users", who are interested in the overall reduction of the energy cost of their computation, which includes both their PC and the Cloud infrastructure. For the sake of simplicity, we have not considered energy costs introduced by the network connection to the cloud. The two terms may make sense from a societal angle where human users may be interested in reducing their own devices' energy consumption only or may care about reducing the total environmental impact of their computation, but they can also have practical technical meaning from a system perspective. For instance, an energy-selfish entity could be a battery-operated device, such as a vehicle, a wearable device or a sensor, which for operational reasons is designed to offload its computation to a cloud infrastructure that is not resource-constrained.

For energy-selfish users, we have observed that offloading is most beneficial for the application that runs on a single core, as the local power consumption dropped significantly without a noticeable increase in execution time. The other three applications also experienced considerable reduction in local power consumption, but mostly at a noticeable expense in execution time. Overall, all applications have considerably reduced local energy usage when offloaded (varying from 63.75 % up to 98.88 % reduction in energy introduced by the application compared to local execution). For energy-altruistic users, we have also observed that offloading clearly benefits the single-core application, since, again, the execution time does not increase much, but for the rest of the applications executing them remotely significantly increases the total energy of the system, simply because the energy costs for running a cloud are much higher. Looking at the applications in isolation though, the total amount of energy introduced by each one is less for remote execution (varying from 0.97 % up to 20.28 %) compared to local execution.

5 Conclusions and Future Work

This paper has studied the viability of computation offloading for PCs with respect to the energy *Vs.* performance tradeoff for computationally heavy applications. We see that in most cases, the user can sacrifice performance to make considerable energy savings, not only locally, but also when the total energy cost, including the cloud's, is taken into account. If a cloud infrastructure already exists and runs applications, adding one more incurs less total energy cost at the PC and cloud than a new application would incur running on the PC only. This is significant because it shows that adopting cloud computing can be a meaningful option for reducing the overall carbon footprint of IT. For energy-selfish users, only interested in reducing their own carbon footprint, these savings are considerably greater. In both cases, the energy savings come at the expense of performance. In our experiments, the execution time increase ranged between

1 % and 84 % depending on the application. These initial experiments have provided a valuable baseline for exploration and we plan to extend them for different VM configurations. Looking at other areas of future work, we will investigate simultaneous executions of many computationally light applications. This will yield more accurate relation between the amount of energy saved and other factors like computation power of the cloud and the heaviness of applications that are offloaded.

References

1. Kumar, K., Liu, J., Lu, Y.H., Bhargava, B.: A survey of computation offloading for mobile systems. Mob. Netw. Appl. **18**(1), 129–140 (2013)
2. Gelenbe, E., Lent, R.: Energy-QoS trade-offs in mobile service selection. Future Internet **5**(2), 128–139 (2013)
3. Rahimi, M.R., Ren, J., Liu, C.H., Vasilakos, A.V., Venkatasubramanian, N.: Mobile cloud computing: a survey, state of art and future directions. Mob. Netw. Appl. **19**(2), 133–143 (2014)
4. Othman, M., Madani, S.A., Khan, S.U.: A survey of mobile cloud computing application models. IEEE Commun. Surv. Tutorials **16**(1), 393–413 (2014)
5. Gelenbe, E., Lent, R.: Optimising server energy consumption and response time. Theor. Appl. Inform. **24**(4), 257–270 (2012)
6. Sakellari, G., Loukas, G.: A survey of mathematical models, simulation approaches and testbeds used for research in cloud computing. Simul. Modell. Pract. Theor. **39**, 92–103 (2013)
7. Kumar, K., Lu, Y.H.: Cloud computing for mobile users: can offloading computation save energy? Computer **43**(4), 51–56 (2010)
8. Gelenbe, E., Lent, R., Douratsos, M.: Choosing a local or remote cloud. In: Network Cloud Computing and Applications, pp. 25–30 (2012)
9. Xia, F., Ding, F., Li, J., Kong, X., Yang, L.T., Ma, J.: Phone2Cloud: exploiting computation offloading for energy saving on smartphones in mobile cloud computing. Inf. Syst. Front. **16**(1), 95–111 (2014)
10. Kemp, R., Palmer, N., Kielmann, T., Bal, H.: Cuckoo: a computation offloading framework for smartphones. In: Gris, M., Yang, G. (eds.) MobiCASE 2010. LNICSSITE, vol. 76, pp. 59–79. Springer, Heidelberg (2012). doi:10.1007/978-3-642-29336-8_4
11. Gordon, M.S., Jamshidi, D.A., Mahlke, S., Mao, Z.M., Chen, X.: COMET: code offload by migrating execution transparently. In: Proceedings of the USENIX Symposium on Operating Systems Design and Implementation, pp. 93–106 (2012)

12. Cuervo, E., Balasubramanian, A., Cho, D.K., Wolman, A., Saroiu, S., Chandra, R., Bahl, P.: MAUI: making smartphones last longer with code offload. In: Proceedings of ACM Mobile systems, applications, and services, pp. 49–62 (2010)
13. Geng, Y., Hu, W., Yang, Y., Gao, W., Cao, G.: Energy-efficient computation offloading in cellular networks. In: IEEE ICNP, pp. 145–155 (2015)
14. Shi, C., Habak, K., Pandurangan, P., Ammar, M., Naik, M., Zegura, E.: Cosmos: computation offloading as a service for mobile devices. In: Proceedings of ACM MobiHoc, pp. 287–296 (2014)
15. Zhang, W., Wen, Y., Wu, D.O.: Energy-efficient scheduling policy for collaborative execution in mobile cloud computing. In: IEEE INFOCOM, pp. 190–194 (2013)